DIY Vis Applications: Custom Visualization Applications with VisTrails, VisMashups, and crowdLabs

Emanuele Santos and David Koop
SCI Institute, University of Utah
Agenda

- Context and Requirements
- The VisTrails System
- Custom Visualization Applications in VisTrails
  - Creating Apps
    - Using VisTrails packages and custom widgets
    - VisMashups
  - Sharing Apps
    - crowdlabs.org
Online

- VisTrails: www.vistrails.org
- crowdLabs: www.crowdlabs.org

- This session:
  - http://www.vistrails.org/index.php/VisMashup
Context and Requirements

- Provenance
- Weave multiple libraries
- Access to High-Performance Computing Resources

- Different communities want their customized GUI
- Test results using an ordinary Web browser
- Control who can access the information
- Tools for sharing and publishing results need to be integrated with the tools the scientists already use
Provenance in Art

Rembrandt van Rijn
Dutch, 1606 - 1669

Self-Portrait, 1659
oil on canvas
Andrew W. Mellon Collection
1937.1.72

Provenance
George, 3rd Duke of Montagu and 4th Earl of Cardigan [d. 1790], by 1767;[1]
by inheritance to his daughter, Lady Elizabeth, wife of Henry, 3rd Duke of
Buccleuch of Montagu House, London; John Charles, 7th Duke of Buccleuch;
(P. & D. Colnaghi & Co., New York, 1928); (M. Knoedler & Co., New York); sold
January 1929 to Andrew W. Mellon, Pittsburgh and Washington, D.C.; deeded
28 December 1934 to The A.W. Mellon Educational and Charitable Trust,
Pittsburgh; gift 1937 to NGA.

[1] This early provenance is established by presence of a mezzotint after the
portrait by R. Earlom (1743-1822), dated 1767. See John Charrington, A
Catalogue of the Mezzotints After, or Said to Be After, Rembrandt, Cambridge,
1923, no. 49.

Associated Names
• Buccleuch, Henry, 3rd Duke of
• Buccleuch, John Charles, 7th Duke of
• Colnaghi & Co., Ltd., P. & D.
• Knoedler & Company, M.
• Mellon, Andrew W.
• Mellon Educational and Charitable Trust, The A.W.
• Montagu, and 4th Earl of Cardigan, George, 3rd Duke of
Provenance in Science

- Provenance is as (or more) important as the result!
- Old solution: Lab notebooks
- New problems:
  - Large volumes of data
  - Complex analyses
  - Writing notes doesn’t scale
- New solution:
  - Automated provenance capture with user-defined annotations

[DNA Recombination, Lederberg]
Provenance in Visualization

**Provenance Model**

**Efficient Storage and Querying**

**Systematic Capture**

**Usable Tools**

Fig. 4: Visualizing a binary star system simulation. This is an image figure directly in the article. The original workflow is available at http://www.swivel.com.

Fig. 7: Using the blog to document processes, we created a series of blog posts to explain the problems found when generating a binary star system simulation. Using the blog to document processes enables the visualizations for MOP.

Fig. 6: Visualizing a binary star system simulation. This is an image figure directly in the article. The original workflow is available at http://www.swivel.com.

ACKNOWLEDGMENTS

Our research has been funded by the National Science Foundation grants: IIS-0845862, IIS-0750369, ATM-0638525, IIS-064456, CNS-0571532, IIS-0735637, OCI-0824962, IIS-0748429, CNS-0514485, IIS-0719302, CNS-0226666, CCF-0416498, OCS-0203587, OCS-0213002, and OCS-0213003, with support from the Data Visualization and Scientific Computing initiative at the Department of Energy, the National Science Foundation, and the International Provenance and Annotation Workshop (IPAW).

REFERENCES


The VisTrails System

- Comprehensive provenance infrastructure for computational tasks
- Focus on exploratory tasks such as simulation, visualization, and data analysis
- Transparently tracks provenance of the discovery process—-from data acquisition to visualization
  - The trail followed as users generate and test hypotheses
- Leverage provenance to streamline exploration
- Focus on usability—build tools for scientists
Specifications

• Open-source, freely downloadable system (www.vistrails.org)
• Multi-platform: users on Mac, Linux, and Windows
• Python code and uses PyQt and Qt for the interface
• Over 20,000 downloads in two years
• User’s guide, wiki, and mailing list
• Many users in different disciplines and countries:
  • Visualizing environmental simulations (CMOP STC)
  • Simulation for solid, fluid and structural mechanics (Galileo Network, UFRJ Brazil)
  • Quantum physics simulations (ALPS, ETH Switzerland)
  • Climate analysis (CDAT)
  • Habitat modeling (USGS)
  • Open Wildland Fire Modeling (U. Colorado, NCAR)
  • High-energy physics (LEPP, Cornell)
  • Cosmology simulations (LANL)
  • Using tms for improving memory (Psychiatry, U. Utah)
  • eBird (Cornell, NSF DataONE)
  • Astrophysical Systems (Tohline, LSU)
  • NIH NBCR (UCSD)
  • Pervasive Technology Labs (Heiland, Indiana University)
  • Linköping University
  • University of North Carolina, Chapel Hill
Climate Data Analysis

[UVCDAT Project, Lawrence Livermore National Lab]
Quantum Lattice Models

[ALPS Project, ETH-Zurich]
Coastal Margin Observation & Prediction
Comparing Cosmological Simulations

[Cosmic Code Comparison Project, Los Alamos National Lab]
VisTrails Change-based Provenance

- Undo/redo stacks are linear!
- We lose history of exploration
- Old Solution: User saves files/state
- VisTrails Solution:
  - Automatically & transparently capture entire history as a tree
  - Users can tag or annotate each version
  - Users can go back to any version by selecting it in the tree
Change-based Provenance

[Freire et al., IPAW 2006]
Change-based Provenance

Change 1 (add module):
add module MplPlot
Change 2 (change configuration):
add function source("vspr = self.getInputFromPort(...)"
Change 3 (add connection):
add connection vtkStructuredPointsReader → MplPlot
Change 4 (paste):
add module MplFigure
add module MplFigureCell
add connection MplFigure → MplFigureCell
Change 5 (add connection):
add connection MplPlot → MplFigure

\[ vt = x_n \circ x_{n-1} \circ \ldots \circ x_1 \circ \emptyset \]

where each \( x_n \) is a change

[Freire et al., IPAW 2006]
Execution Provenance

<module id="12" name="vtkDataSetReader"
  start_time="2010-02-19 11:01:05"
  end_time="2010-02-19 11:01:07">
  <annotation key="hash"
    value="c54bea63cb7d912a43ce"/>
</module>

<module id="13" name="vtkContourFilter"
  start_time="2010-02-19 11:01:07"
  end_time="2010-02-19 11:01:08"/>

<module id="15" name="vtkDataSetMapper"
  start_time="2010-02-19 11:01:09"
  end_time="2010-02-19 11:01:12"/>

<module id="16" name="vtkActor"
  start_time="2010-02-19 11:01:12"
  end_time="2010-02-19 11:01:13"/>

<module id="17" name="vtkCamera"
  start_time="2010-02-19 11:01:13"
  end_time="2010-02-19 11:01:14"/>

<module id="18" name="vtkRenderer"
  start_time="2010-02-19 11:01:14"
  end_time="2010-02-19 11:01:14"/>
Hands-On Part 1
VisTrails for Vis Applications

- Adding packages to VisTrails
- Build GUIs on top of VisTrails
- VisTrails plug-ins
Adding VisTrails packages

- seawater python package:
  - http://pypi.python.org/pypi/seawater/1.0.3

```
identifier = 'org.ocefpaf.seawater'
version = '1.0.3'
name = 'Seawater Routines'

import seawater

class SaturationN2(Module):
    _input_ports = [(['S', Float],
                     ['T', Float])]
    _output_ports = [(['res', Float])]

def compute(self):
    s = self.getInputFromPort("S")
    t = self.getInputFromPort("T")
    res = seawater.satN2(s, t)
    self.setResult('res', res)

_modules = [SaturationN2,
```
GUls in VisTrails
GUls in VisTrails

Interacting with the UV-CDAT window or shell automatically generates provenance

[UVCDAT Project, Lawrence Livermore National Lab]
VisTrails Components

Workflow Editor

Version History

Visual Spreadsheet

Workflow Execution
VisTrails Maya Plug-in

[VisTrails, Inc., IPAW 2008]
Provenance Enabling 3rd-Party Tools

- Autodesk Maya
- ParaView
- VisIt
- ImageVis3d
Context and Requirements

- Provenance
- Multiple libraries
- Access to High-Performance Computing Resources

- Different communities want customized GUIs
- Test results using an ordinary Web browser
- Control who can access the information
- The tools for sharing and publishing results need to be integrated with the tools the scientists already use
Sharing Analysis

- Visualizations are hard to share and reuse
  - Powerpoint files, screenshots, videos
  - File dependencies
  - Tools that are general and complex
  - Often code is not multi-platform
- Visualization portals (such as Many Eyes)
  - Limited to a set of predefined visualizations
  - Do not support private data
VisMashup: Simplifying the Creation and Deployment of Custom Visualization Applications

- Create and modify visualization applications more easily

[Santos et al., Vis 2009]
VisMashup

- Simplifies the creation, maintenance, deployment and use of customized visualization applications (mashups)
- Uses dataflows as the underlying model
- Keeps detailed provenance information of the application development process and use
VisMashup Overview

1. Acquire and Analyze Pipelines
2. Create Views (Simplify Pipelines)
3. Combine Views
4. App generation and deployment
Acquire and Analyze Pipelines
Create Views (Simplify Pipelines)
Combine Views
App generation and deployment

Mine pipeline provenance

VisMashup - M...
• Deploy visualization applications on different configurations.
crowdlabs.org

- Facilitates collaboration and sharing among scientists and visualization experts
- Adopts Social Web Sites Model
- Integrates a set of usable tools and a scalable infrastructure
- Leverages provenance information
Hands-On part 2

- Preparing workflows in VisTrails
- Uploading workflows to crowdlabs.org
- Creating vismashups
- Uploading vismashups to crowdlabs.org
Preparing workflows in VisTrails

- Attention to how the workflow accesses data files
  - If you want your workflows to be runnable by other users, they should use HTTPFile or RepoSync modules
- Create aliases for parameters to be exposed
Uploading workflows to crowdlabs.org

- Login to your account on the repository
Uploading workflows to crowdlabs.org

- Set up permissions and push the vistrail to the repository
Creating Vismashups

• Press Show Medley Editor button
Creating Vismashups

- Press Create a Pipeline View button (Pipeline Views panel)
Creating Vismashups

• Load a head_mashup vistrail from crowdlabs.org
Creating Vismashups

- Load a `head_mashup` vistrail from crowdlabs.org
Creating Vismashups

- Select histogram node
Creating Vismashups

- Select isovalue and add some sample values to Values List
Creating Vismashups

• Give a name to your Pipeline View
Creating Vismashups

- Preview App: drag histogram to canvas and press Execute
Creating Vismashups

• Desktop App
Uploading Vismashup to crowdlabs.org
Uploading Vismashup to crowdlabs.org
Summary and Conclusions

• A framework for simplifying the creation and deployment of collaborative data analysis and visualization tools

• VisMashup:
  • Simplifies the creation of custom visualization applications
  • Developers can quickly assemble custom applications, leveraging an existing collection of visualization pipelines and their provenance
  • Generated apps are flexible and simple enough for end-users

• CrowdLabs:
  • Facilitates collaboration and sharing among scientists and visualization experts
  • Scientists can create and publish new data products with minimal intervention of expert programmers and IT personnel
Future Work

• Use the provenance captured by VisMashup to improve GUI generation
• Use distributed model for workflow execution that depends on the client configuration (for portable devices)
• Advanced Provenance Analytics
  • Workflow search engine
  • Recommendations based on the file information
  • Connect people based on the types of visualizations they are interested in, etc.
• Improve Web-enabled interfaces and graphics (with 3D support)
• Streaming and multi-resolution techniques for supporting remote data
Thank you!

• Questions?